Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 15 (2014) 376 – 387

Procedia Technology

2nd International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering

Sequential dynamic optimization of complex nonlinear processes based on kriging surrogate models

Ahmed Shokry[a], Antonio Espuña[a]*

[a]*Department of Chemical Engineering Universitat Politecnica de Catalunya Av. Diagonal, 020280, Barcelona, Spain.*

Abstract

This paper presents a sequential dynamic optimization methodology applicable to solve the optimal control problem of complex highly nonlinear processes. The methodology is based on the use of kriging metamodels to obtain simpler, accurate, robust and computationally inexpensive predictive dynamic models, derived from input/output (training) data eventually generated using the original complex first principles process model (mathematical or analytical model) or from the real system. Then these metamodels can easily take the place of the complex first principles process model in any of the well-tailored computational schemes of sequential dynamic optimization. The results of applying this approach to three well known problems from the process systems engineering area are compared with the ones obtained using the corresponding first principles models, showing how the proposed approach significantly reduces the computational effort required to get very accurate solutions, and so enables the use of dynamic optimization procedures in applications where robustness and immediacy are essential practical constraints.

Peer-review under responsibility of the Organizing Committee of SysInt 2014.

Keywords: Optimal control; dynamic optimization; surrogate based optimization; kriging; system identification

1. Introduction

A key element to improve system performance in the process industry (e.g.: to reduce the operating cost, to increase the production yield, or to ensure product quality) is the fast and reliable identification of the adequate time

* Corresponding author. Tel.: +34 934011732.
E-mail address: antonio.espuna@upc.edu

Peer-review under responsibility of the Organizing Committee of SysInt 2014.
doi:10.1016/j.protcy.2014.09.092

profiles to be followed by the process control variables (e.g.: equipment unit feed rates, cooling temperature profiles, etc.). In most cases, the optimum profiles are scenario dependent (i.e.: the profile must be adapted according to the quality requirements, the characteristics of the raw materials, the economic conditions, etc.); additionally, a large number of uncertain variables must be often contemplated, and the relations between the control variables and the performance are usually difficult to model in detail. In such cases, a priori calculations (parametric optimization) are not helpful and the process engineer must periodically (even continuously) solve the associated model based control problem (open loop optimal control [1]), which requires going through a complex mathematical procedure (dynamic optimization) which involves a dynamic model (differential equations), a multifaceted objective (usually based on the final state of the system but also on its evolution), and a set of control variables which eventually can change at any time.

State-of-the-art methods for solving dynamic optimization problems of industrial relevance rely on the application of the so-called Direct Methods [1], based on the discretization of the time domain and the transformation of the original infinite continuous optimal control problem into a finite constrained nonlinear programming (NLP) problem, which is then solved by appropriate numerical nonlinear optimization tools (e.g. Sequential Quadratic Programming (SQP), Trust Region Search…etc.). Alternatively, Indirect Methods use the analytical necessary conditions from the calculus of variations to formulate a boundary value problem, which is usually very difficult to solve [1] and requires a deep a priori knowledge of the nature of the problem (initialization, constraints structure, etc.), so they are usually inapplicable to the industrial practice.

Direct methods are further classified according to the elements finally discretized: sequential approaches (also known as Control Vector Parameterization (CVP) approaches) discretize only the control variables in the form of piecewise low order polynomials, and then a NLP optimization problem is carried out in the space of the discretized control variables, which requires the successive evaluation (simulation runs) of the nonlinear process model during its solution. On the contrary, simultaneous approaches discretize both control and state variables by approximating them by a family of polynomials on finite elements [2], so they avoid the inner evaluation of the differential process model, although they result in a NLP problem of a very large-scale (due to the presence of state variables together with the control variables as optimization variables [1, 4]) and require the introduction of extra constraints to enforce the continuity of the discretized state variables [7].

The sequential strategy is straightforward and relatively easy to construct and to apply, and results in a NLP optimization problem of a much reduced size [1,2,4,7]. However, a major challenge that faces the sequential approach is the huge computational effort required, associated to a large number of evaluations of the nonlinear process model, since each evaluation includes the expensive integration of this differential model using complicated integration techniques [2,7]. This challenge is amplified in case of complex, highly nonlinear problems (e.g. chemical processes) [2,7,14], and the computational cost may become unaffordable if a fast identification of the process control profiles is required, which is the case in many industrial applications (e.g. transitions between desired operating conditions, response to sudden disturbances or unexpected events, online optimization - model based control, etc.) [2,13].

In these cases, one solution is to use inexpensive accurate surrogate models (e.g. artificial neural networks), based on the input/output simulation data obtained through the solution of a complex model [9,13] usually based on first principles. Kriging metamodels outperform many other metamodel types in many engineering fields, because of their specific properties, as high prediction accuracy with a relatively small number of training data, and specially their ability to estimate a prediction variance (or error) which represents the prediction uncertainty [9,10]. However, the majority of kriging usage and developments are concentrated in emulating complex "static" models [8, 9, 10], while engineering systems are of dynamic nature, and so the use of dynamic models is a must in any control application.

In this work, a framework for the application of sequential dynamic optimization techniques to complex industrial processes and systems using ordinary kriging surrogate models is proposed. First, a strategy for using ordinary kriging metamodels to obtain simple and accurate dynamic predictive models of nonlinear processes/systems is presented. Second, a typical sequential dynamic optimization procedure is tuned to integrate a set of dynamic kriging metamodels representing a complex process model. Finally, a detailed description of the steps conforming the proposed methodology and its application to benchmark case studies in chemical industries are presented.

2. Methodology techniques and tools

A dynamic system is characterized through a set of state variables x_t, evolving from their initial values x_0 over time horizon $[t_0: t_f]$, and being affected by the system inherent dynamics, and a set of control variables u_t, which can be externally manipulated within a certain range during this time horizon, affecting the inherent system dynamics. So the objective is to find the profile of the control variables to obtain the best value of a certain objective function.

2.1. Sequential dynamic optimization

The direct sequential approach to solve dynamic optimization problems is based on the discretization or parameterization of the control variables u_t as piecewise polynomials† $[u_1, u_2, u_3 \dots, u_{N-1}, u_N]$ [1,7] by dividing the total time domain $[t_0 : t_f]$ into a grid of N equally sized intervals Δt, where $\Delta t = t_j - t_{j-1} = t_f - t_0/N, j=1,2,\dots N$, where $[t_0 < t_1 < t_2 < \dots\dots < t_{N-1} < t_N = t_f]$. The optimization is carried out in the space of the parameterized control variables u_j only, which became decision variables. In each iteration, the NLP solver updates the values of the discretized control variables u_j, the differential process model (2) is integrated using standard integration algorithms (e.g. Runge-Kutta), the state variables x_t are calculated from the known initial conditions x_0, and the objective function J (1) and the constraints g (3) are evaluated [1,2,7].

$$\underset{x(t),u_j(t)}{Min}\ J = \Phi\left(x(t_N)\right) + \int_{t_0}^{t_f} \varphi(x(t),u_j(t),t)\,dt \tag{1}$$

$$s.t.\quad \frac{\partial x}{\partial t} = F(x(t),u_j(t)) \qquad ,j = 1,2...N, \quad x(t_0) = x_0 \tag{2}$$

$$g(x(t),u_j(t)) \le 0 \qquad ,j = 1,2,....N \tag{3}$$

$$u_{\min} \le u_j(t) \le u_{\max} \tag{4}$$

2.2. Kriging metamodel construction, parameter estimation, and validation

The construction of an accurate surrogate model relies on the representativeness of the available training points (input/output simulation data). Whenever it is feasible, training points should be selected in such a way that the best representation of the original model behavior is obtained. This selection task is called the "sampling plan design" [8], and it results in a set of input combinations (sampling plan $[W]_{n\times k}$, where n is the number of sample points, and k is the number of input variables w) at which the corresponding output data (response variables - output data matrix $[Y]_n$) are expected to be obtained [3]. The Space-Filling Latin Hypercube Sampling design (SLHS) has been used in this work as sample plan design technique [10].

The ordinary kriging assumes a stochastic process, in which the error in the predicted value is also a function of the input variables w. The kriging predictor $\hat{y}(w)$ is then composed by two parts (5): the first one is a polynomial term $f(w)$ which is constant $f(w)=\mu$; the second part is the residual term $Z(w)$ from that polynomial, where $Z(w)$ is a stochastic Gaussian process that represents the uncertainty about the mean of $\hat{y}(w)$, with an expected zero value $E(Z(w))=0$, and a covariance between two points (w_i, w_j) calculated as: $cov(Z(w_i),Z(w_j)) = \sigma^2 R(w_i,w_j)$, where σ^2 is the process variance, and $R(w_i,w_j)$ is the spatial correlation function (6), which is usually selected exponential [3, 9].

$$\hat{y}(w) = f(w) + Z(w) \tag{5}$$

† For simplicity, these polynomials are usually assumed to follow piecewise constant profiles. This has been also the choice in this paper.

$$R(w_i, w_j) = \exp\left(-\sum_{l=1}^{k} \theta_l \left| w_{i,l} - w_{j,l} \right|^{p_l} \right) \quad l = 1, 2, ...k \tag{6}$$

To estimate the values of μ, σ^2, θ_l, P_l, the likelihood function of the observed data [Y] should be maximized (the smoothness parameters P_l are usually kept to the value of 2, which provide smooth infinitely differentiable correlation function). The optimal values for μ and σ^2 can be expressed as indicated by equations (7) and (8) (see [8] for a detailed derivation of these expressions). The kriging predictor (9) and its variance (10) can be obtained by calculating the likelihood function of the original data set augmented with the new interpolating point (w_{new},y_{new}) , where r is the $n \times 1$ vector of correlations $R(w_{new}, w_i)$ between the point to be predicted w_{new} and the sample design points.

$$\hat{\mu} = \frac{1^T R^{-1} Y}{1^T R^{-1} 1} \tag{7}$$

$$\hat{\sigma}^2 = \frac{\left(Y - 1\mu\right)^T R^{-1} \left(Y - 1\mu\right)^T}{n} \tag{8}$$

$$\hat{y}(w_{new}) = \hat{\mu} + r^T R^{-1} \left(Y - 1\mu\right) \tag{9}$$

$$\hat{S}^2(w_{new}) = \hat{\sigma}^2 \left(1 - r^T R^{-1} r + \frac{\left(1 - r^T R^{-1} r\right)^2}{r^T R^{-1} r} \right) \tag{10}$$

Finally, the metamodel should be checked that it exhibits a satisfactory range of accuracy within its domain [3]. Cross validation allows the characterization of the metamodel error without any additional data rather than the original set of sample points [12]. Various techniques of cross validation have been developed, as the "P-fold cross validation" and the "leave-K-out cross validation". In this paper the "leave-one-out cross validation" (LOOCV) is used: in each iteration, one sample point is held out for validation, and the remaining points are used to fit the metamodel; the cross validation error at each iteration is calculated, and the average root mean square error of the cross validation of the surrogate model is calculated from equation (11), where n is the total number of sample points, and $\hat{y}_i$, y_i are the estimated and the real value- respectively -of the held out point (w_i), [12].

$$AVR_{\left\{RMSE_{\{cv\}}\right\}} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(\hat{y}_i - y_i \right)^2} \tag{11}$$

3. Kriging based dynamic modeling and simulation methodology

In the case of a dynamic system, the kriging metamodel(s) must be trained to mimic the incremental evolution of the dynamic system behavior (state variables), over a relatively small and constant time step Δt. In this case, the metamodel output $\hat{y}$ in equation (5) will be composed by the state variables x_j at the time t_j ($\hat{y} = x_j$), and the metamodel input variables w will include both the control variables u_j and the state variables at the previous time x_{j-1} ($w = [u_j, x_{j-1}]$). Hence, the dynamic kriging metamodel (which will be further referred as "Constant Step Dynamic Kriging" - CSDK) will be given by equation (12) .

$$\hat{x}_j = f\left(x_{j-1}, u_j\right) \quad , \Delta t = t_j - t_{j-1} \tag{12}$$

Assuming that an accurate (but complex) model of the system is available (e.g.: a first principle model, adequately tuned), A CSDK model can be easily derived following the idea of using "computer experiments" [9,10] to generate the training data: As a first step, the range (domain) within which the input variables (the state variables x_{j-1} and the control variables u_j) are expected to move is estimated [$x_{t\text{-}min} : x_{t\text{-}max}$, $u_{t\text{-}min} : u_{t\text{-}max}$]. Then a SLHS sampling plan $[W]_{n\times k}$ is designed over this expected domain. At each row (point) of the sampling plan, a computer experiment (simulation run) is carried out using the original complex first principle model over a fixed and relatively small time step Δt, to obtain the corresponding outputs $[Y]_{n\times M}$, where M is the number of outputs (state variables). After fitting the M kriging metamodels (one kriging model for each state variable), the CSDK metamodels are validated using the LOOCV, to ensure that they possess acceptable range of prediction accuracy.

Once obtained accurate CSDK model(s), it/they can be then used in a recursive way to predict or interpolate the entire time series/sequence of outputs $[\hat{x}_1, \hat{x}_2, \hat{x}_3, ..., \hat{x}_{N-1}, \hat{x}_N]$ (the dash on the x means it is an estimated value). The recursive dynamic interpolation or emulation starts using the given input values (x_0, u_1) to interpolate or predict $\hat{x}_1$, then with ($\hat{x}_1$, u_2) to predict $\hat{x}_2$, then with ($\hat{x}_2$, u_3) to predict $\hat{x}_3$, and so on, until the last interpolation step using ($\hat{x}_{N-1}$, u_N) to predict $\hat{x}_N$.

To illustrate the capability of the CSDK models to emulate a nonlinear dynamic process for large time domain, we have applied the technique to a tank draining system, which is a well-known example that has been commonly used in control text books and software. The system model (13) has one state variable $x(t)$, which represents the water height inside the tank, and one control variable $u(t)$ which corresponds to the inlet flowrate entering the tank. The water leaves the tank through the bottom part, under the gravity effect. b and a are constants related to the inlet and outlet flowrates respectively, and A represents the cross section area of the tank. Then, the first principle model representing the systems dynamics can be expressed as indicated by equation (13).

$$\frac{dx}{dt} = \frac{1}{A}(b\,u - a\,x^{0.5}) \tag{13}$$

Since there is one state variable, only one CSDK model is required to emulate the system. The input variable [x_{j-1}, u_j] is expected to vary within the limits of [*0: 4, 0: 1.5*]. Over this domain, a SLHS plan is designed with *65* sample points. The original model in equation (13) is used to generate the output matrices [x_j], Δt=*0.5 min*. The metamodel fitting is achieved through the maximization of likelihood of the observed data, and the obtained values of the metamodel parameters are (μ, σ^2, [θ_l]) = (*2.24* , *0.48*, [*8.1*, *1.4*]).

The accuracy of the fitted CSDK model is assessed by the LOOCV technique, through equation (11). The metamodel cross validation results using the LOOCV method is shown in Fig. 1(a), and the average root mean square error of the cross validation is obtained ($AVR_{RMSE} = 8.13\times10^{-4}$). The extremely low value of the AVR_{RMSE} (almost 4 orders of magnitude less than the expected values range) indicates that the fitted CSDK model exhibits a very high accuracy, and it is ready to be used for dynamic emulation:

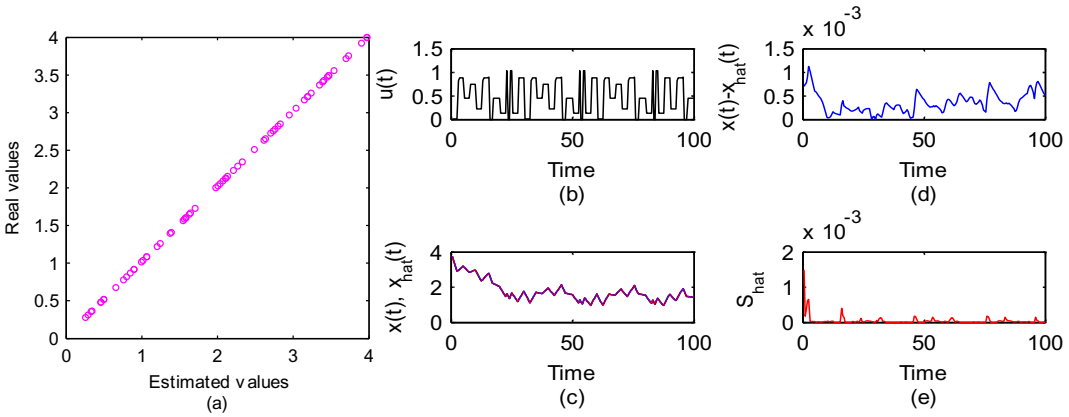

Fig. 1. Water tank simulation using the CSDK model: (a) LOOCV of the CSDK model, (b) Control profile to be emulated, (c) Simulated dynamic behavior using the mathematical model (blue) $x(t)$, and the CSDK $x_{hat}(t)$ (red dashed line), (d) Absolute error between the real and the estimated dynamic responses, (e) Estimated error of the CSDK model.

Fig. 1(c) shows the comparison between the dynamic simulation of a randomly generated control profile (200 time step ($200\times\Delta t$ =100 *min*) Fig. 1(b)), using the fitted CSDK model (red dashed line), and the model based on first principles(13) (solid blue line), both departing from an initial value of the state variable of x_0=3.96 *m*. Both lines correspond to virtually identical values, so it is clear that the CSDK is able to accurately estimate the same behaviour of the dynamic system. Fig. 1(d) plots the absolute error between the kriging prediction and the values calculated by the first principles model. It is also worth to compare Fig. 1(d) with Fig. 1(e), which represents the kriging estimated error (10): this estimated error in maximum in the same situations where the observed difference has been found also maximum.

4. Kriging based control vector parameterization

The previous steps and techniques have been used to construct a robust computational framework for the application of CVP based dynamic optimization:

1. Explore the process behavior and identify the state variables x_t to be modeled and the control variables u_t.
2. Discretize the time domain $[t_0: t_f]$ into a grid of equal time steps $[t_0<t_1<t_2< \ldots\ldots<t_{N-1}<t_N=t_f]$, and discretize the control variables u_t as piecewise constants $[u_1, u_2, u_3 \ldots, u_{N-1}, u_N]$
3. Estimate the range of the state and control variables $[x_{t\text{-}min} : x_{t\text{-}max}, u_{t\text{-}min} : u_{t\text{-}max}]$.
4. Design a sampling plan $[W]_{n\times k}$ over the metamodel(s) domain, using the SLHS technique.
5. Carry out a simulation run at each point of the sampling plan using the original (complex) model in order to obtain the corresponding response states $[Y]_{n\times M}$.
6. Fit the CSDK model(s) and validate it/them as described (LOOCV),
7. Integrate the CSDK model(s) in the CVP dynamic optimization scheme:

    a. Determine an initial guess for the decision variables $u_0=[u_{01}, u_{02} \ldots\ldots u_{0N-1}, u_{0N}]$.
    b. Integrate the dynamic system state variables until the final time horizon, using the CSDK model(s), and compute the performance index J and the constraints g.
    c. Use a NLP optimizer (e.g SQP, trust region search) to update the values of the parameterized control variables, until the objective function is minimized, and the optimal control u^* policy is obtained.

5. Application and results

The proposed kriging based CVP methodology is applied to three benchmark problems, which are commonly used in dynamic optimization studies. The methodology is applied to each case study two times, each one with a different discretization of the total time domain (15 and 20 time steps discretization).

In each discretization, the methodology results are compared with the use of the classical or standard CVP technique, with the same problem adjustment (optimization domain, time steps, initial guess of the control profile, NLP optimizer, etc.), but using the first principles mathematical model, integrated using the Matlab ODE algorithm *ode15s*. The first case is explained in detail; the application to the other two cases is straightforward so just the main results are commented. Finally, it is worthy to note that, in order to facilitate the comparison of results among the different case-studies, the control policies obtained in the three examples are scaled between [0, 1].

5.1. Case 1:Plug flow reactor catalyst blend problem

In this problem [6], a plug flow reactor is to be packed with a mixture of two different types of catalysts (type *1* and type *2*). The kinetic sequence is given by: $A \leftrightarrow B \rightarrow C$ and the problem is to find the optimal profile of catalyst of type *1* ($u(z)$) along the reactor (in this problem, the independent variable z represents the reactor length, rather than time) , to maximize the production of component C (J, equation (14)).

$$\underset{u}{Max}\ J = 1 - x_A(z_f) - x_B(z_f) \tag{14}$$

$$s.t. \qquad \frac{dx_A}{dz} = u(z)[10x_B(z) - x_A(z)], \qquad \frac{dx_B}{dz} = -u(z)[10x_B(z) - x_A(z)] - [1 - u(z)]x_B(z) \tag{15}$$

where, x_A and x_B are the mole fractions of substances A and B respectively, $[x_A(0), x_B(0)]=[\,1\,,\,0]$, $z_f=12$, $0 \le u(z) \le 1$.

First, the proposed methodology is applied using a 15 step discretization of the length domain [*0: 12*], hence each step $\Delta z=0.8$. Since the process has two state variables, two CSDK metamodels (equations (16) and (17)) are fitted to mimic the system. The input variables [x_{Aj-1}, x_{Bj-1}, u_j] are expected to vary within the limits (metamodel/optimization domain) of [*0 : 1.2*, *0 : 0.1*, *0* : *1*]. Over this domain, a SLHS sampling plan is designed with *91* sample points. The original model (15) is used to generate the output matrix [x_{Aj}, x_{Bj}].

$$\hat{x}_{Aj} = f\left(x_{Ai-1}, x_{Bi-1}, u_i\right), \Delta z = z_j - z_{j-1} \tag{16}$$

$$\hat{x}_{Bj} = f\left(x_{Ai-1}, x_{Bi-1}, u_i\right), \Delta z = z_j - z_{j-1} \tag{17}$$

The accuracy of the fitted CSDK models is evaluated using the LOOCV technique, and the AVR_{RMSE} of the cross validation are listed in Table 1. The obtained CSDK models have been used in the kriging based CVP, in which a sequential quadratic optimizer updates the discretized control profile until obtaining the optimal solution. The same methodology has been applied again with a different discretization of 20 step, hence $\Delta z=0.6$, using the same generated sampling plan (the only difference is that the response variables are generated by carrying out the computer experiments with $\Delta z=0.6$). Finally, both problem instances have been solved using the standard CVP technique in which the original first principles mathematical model is used with the same two discretization configurations (15, 20 times steps). The optimization results and computational effort of the proposed methodology and the standard CVP technique are compared in Table 1, and visualized in Fig. 2.

Table 1. Optimization results and kriging metamodels information of case study 1 (*).

		Optimization results (**)			***Metamodels information***	
		Step Δz	*Objective value*	*Computation time* (s)	AVR_{RMSE} (x_A)	AVR_{RMSE} (x_B)
15 steps	*Frist principles*	12/15=0.8	0.4745	70.60		
	kriging	12/15=0.8	0.4739	16.72	7.5×10^{-5}	8.9×10^{-5}
20 steps	*Frist principles*	12/20=0.6	0.4753	172.9		
	kriging	12/20=0.6	0.4746	36.00	3.42×10^{-5}	1.3×10^{-5}

(*) Kriging optimization results have been finally assessed using the original first principles model to ensure a fair comparison among both methods

(**)An optimal objective value of *0.477* is reported in the literature [6] without specific indication of the required computational effort.

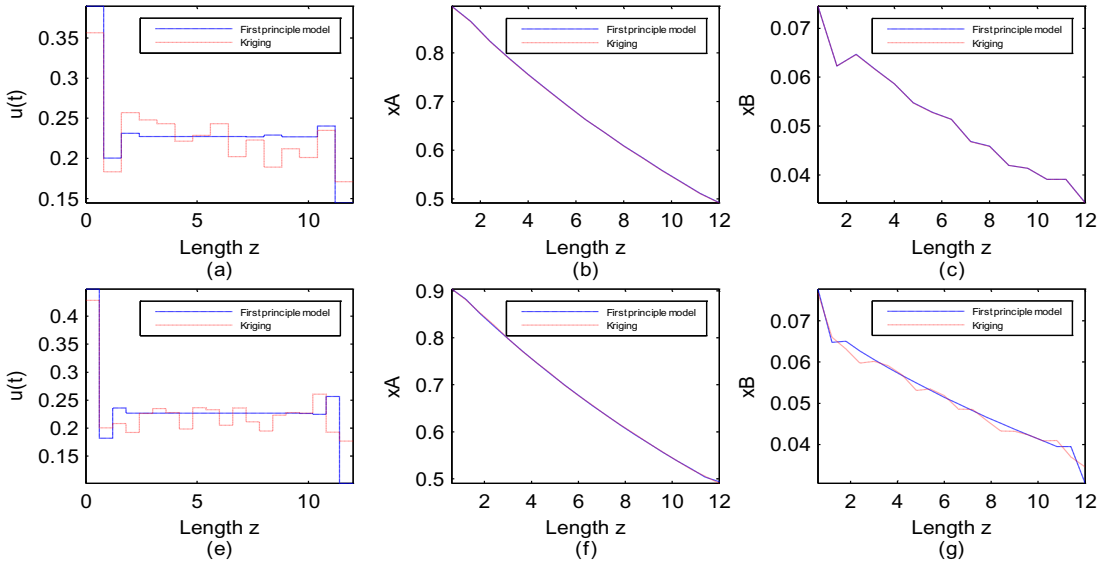

Fig. 2. Optimal control policy and optimal state variables of case (1) using15 step (a,b,c), and 20 step (e,f,g) discretizations: (a,e) Optimal control policies using standard CVP (blue dashed), and proposed methodology (red dotted line), (b,c) and (f,g) State variables of the optimal solutions (standard CVP: blue dashed, kriging: red dotted).

For both discretization settings, the proposed methodology was able to obtain very similar control policies (Fig. 2.(a),(e) red dotted line) to the ones obtained using the first principles process model (Fig. 2.(a),(e) blue dashed line), both leading to almost identical process dynamics (Fig. 2.(b),(c),(f),(g)). Moreover, the methodology has obtained approximately the same global optimal value of the objective (0.13 % difference in case of 15 time steps; 0.14 % in case of 20 time steps), with relative significant reduction in the computational effort (more than 75% savings).

5.2. Case 2:Batch reactor

In a batch reactor [11], a reversible reaction A↔B is taking place. The problem is to find the best temperature control policy $u(t)$, that maximizes the performance index J in equation (18),

$$\underset{u}{Max}\, J = x_2(t_f) \tag{18}$$

$$st: \quad \frac{dx_1}{dt} = (1-x_1)\times k_1 - x_1 \times k_2, \qquad \frac{dx_2}{dt} = 300\times((1-x_1)\times k_1 - x_1\times k_2) - u\times(x_2 - 290) \tag{19}$$

$$k_1 = 1.7536\times10^{5}\ \exp\left(\frac{-1.1374\times10^{4}}{1.9872\times x_2}\right)\quad,\quad k_2 = 2.4885\times10^{10}\ \exp\left(\frac{-2.2748\times10^{4}}{1.9872\times x_2}\right)$$

where $[x_1(0),x_2(0)] = [0, 380]$, $0 < u(t) < 0.36$, $t_f = 5\ min$.

The time domain $[0: 5]$ is discretized ($\Delta t=0.333$, $\Delta t=0.25$) and, in each case, two CSDK metamodels are fitted within the expected limits (metamodel/optimization domain) $[0: 1, 290: 490, 0: 0.36]$. Over this domain, a SLHS sampling plan is designed with *123* sample points. The original model (equation (19)) is used to generate the output matrix $[x_{1j}, x_{2j}]$ and the accuracy of the fitted CSDK models is evaluated using the LOOCV (Table 2). The obtained CSDK models have been used in the kriging based CVP. The results of the proposed methodology and the standard CVP technique are compared in Table 2, and Fig. 3.

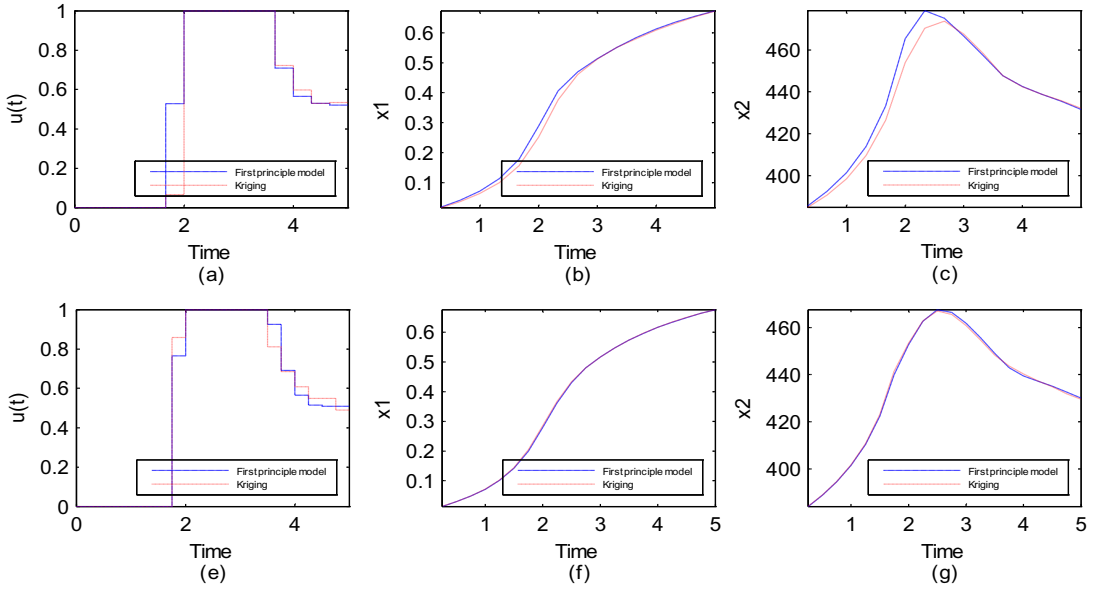

Fig. 3. Optimal control policy and optimal state variables of case (2), using15 time step (a,b,c), and 20 time step (e,f,g) discretizations: (a,e) Optimal control policies using standard CVP (blue dashed), and proposed methodology (red dotted line), (b,c) and (f,g): State variable profiles at the optimal solution (standard CVP: blue dashed, Kriging: red dotted).

Table 2. Optimization results and kriging metamodels information of case study 2 (*).

		Optimization results (**)			***Metamodels information***	
		Time step Δt	*Objective value*	*Computational time* (s)	$AVR_{RMSE}(x_1)$	$AVR_{RMSE}(x_2)$
15 steps	*Frist principles*	5/15=*0.3333*	0.6744	222		
	kriging	5/15=*0.3333*	0.6704	25.5	0.01	3.05
20 steps	*Frist principles*	5/20=*0.25*	0.6749	458		
	kriging	5/20=*0.25*	0.6748	43	0.0031	0.0031

(*) Kriging optimization results have been finally assessed using the original first principles model to ensure a fair comparison among both methods

(**) The reported optimal objective value for this problem in the literature is 0.6753 [11]

For both discretization settings, the proposed methodology is able to obtain a very similar control policy (Fig. 3.(a),(e), red dotted line) to the optimal one obtained from using the real process model (Fig. 3.(a),(e) blue dashed line) and leads to so similar process dynamics (Fig. 3 (b),(c),(f),(g)). Moreover, the methodology has obtained approximately the same objective value (less than 0.6% difference in case of 15 time steps; 0.02 % difference in

case of 20 time steps), with a significant reduction in the required computational effort (about 90 % of the computation time saved in both cases).

5.3. Case 3: parallel reaction problem

In this third problem [6], a tubular reactor is considered to produce two substances according to the parallel reaction A→ B, A→ C, with rate constants k_1 and k_2 respectively. The objective is to maximize the yield of *B* at the final state, finding the most adequate profile of the control variable $u(t)=k_1 l/v$ (l represents the reactor length and v the plug flow velocity).

The dimensionless model describing the system dynamic is given in equations (21), where x_1, x_2 are the dimensionless concentrations of reactants *A* and *B* respectively.

$$\max_u J = x_2(t_f) \tag{20}$$

$$s.t. \quad \frac{dx_1(t)}{dt} = -(u(t) + 0.5\,u^2(t))x_1(t), \qquad \frac{dx_2(t)}{dt} = u(t)x_1(t) \tag{21}$$

where $[x_1(0),x_2(0)] =[1,0]$, $0 <u(t)<5$, $t_f=1$

Again, the dimensionless domain [*0: 1*] is discretized ($\Delta t=0.0667$ and $\Delta t=0.05$) and the proposed methodology is applied and compared with the standard CVP technique. Once the accuracy of the fitted CSDK models is ensured, they replace the complex first principles process model in the CVP. For space limitation, the results of the proposed methodology and the standard CVP technique are compared in Fig. 4 and Table 3 only for the 20 step case.

Table 3. Optimization results and kriging metamodels information of case study 3 (*).

		Optimization results (**)			***Metamodels information***	
		Time step Δt	*Objective value*	*Computational time* (s)	$AVR_{RMSE}(x_1)$	$AVR_{RMSE}(x_2)$
20 steps	*Frist principles*	5/20=*0.05*	0.5733	254.7		
	kriging	5/20=*0.05*	0.5732	34	1.147×10^{-6}	2.14×10^{-6}

(*) Kriging optimization results have been finally assessed using the original first principles model to ensure a fair comparison among both methods

(**) The reported optimal objective value of the problem is *0.5735* [6].

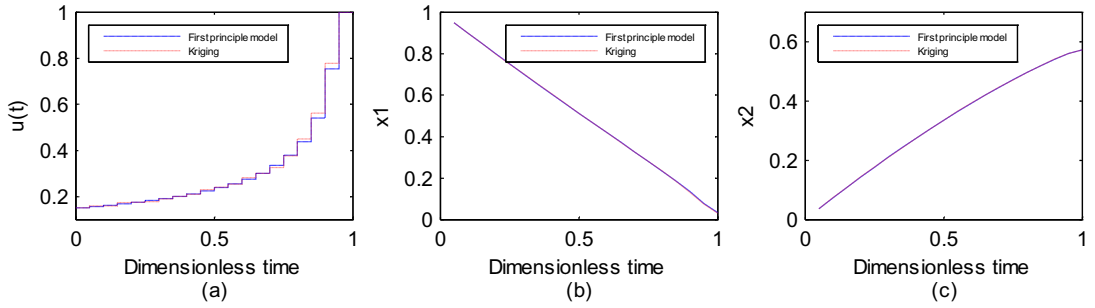

Fig. 4. Optimal control policy and optimal state variables of case (3) using 20 step discretization; (a) Optimal control policies using standard CVP (blue dashed), and proposed methodology (red dotted line), (b) and (c): State variable profiles at the optimal solution (standard CVP: blue dashed; Kriging: red dotted).

The methodology obtained approximately the same global optimal value of the objective with a very small relative error (1.6% and 0.0102%), with relative significant reduction in the computational time (about 85%). In the three case studies, the proposed CVP framework was able to obtain very accurate results. As well as the standard CVP method, the proposed methodology improves the solution using the finer grid (20 time steps), but still saving a huge quantity of computational effort.

6. Conclusion and future work

The potential of dynamic kriging stems from its capacity to replace complex integration rules with simple successive interpolations. This potential has been exploited in this paper to develop a sequential dynamic optimization strategy based on such type of metamodels able to solve the optimal control problem of complex processes, like the ones which usually appear in the chemical or petrochemical sectors, with significant advantages over the use of traditional first principle models. Specifically, relatively simple processes, frequently used as reference in the process systems engineering literature, have been used to assess the eventual benefits of the proposed kriging based CVP strategy in terms of accuracy, robustness and computational cost.

The kriging metamodels have shown high accuracy to capture the nonlinear dynamic nature of these highly nonlinear systems, resulting in an outstanding capacity to predict the system dynamics over large time domains. The integration of such modeling technique with other complementary techniques, like sampling design for computer experiments, cross validation methods and sequential dynamic optimization, to build a comprehensive dynamic optimization framework where the expensive integration of a complex model is replaced by simpler recursive or successive interpolation (CSDK model) has been straightforward. The resulting characteristics and advantages of the proposed framework, and specially its accuracy and the significant reduction of the computational effort, are evident even on its application to three case studies of moderate complexity.

These results confirm that the proposed framework constitutes a significant step forward to solve one of the biggest challenges that face the standard CVP techniques, associated to the significant computational effort required by the repeated process model integration tasks. The methodology becomes also a unique solution when a mathematical process model is missing, and only experimental process data are available. In this sense, it looks like a promising way to allow a more universal application of nonlinear model based control techniques (NMPC), where the optimal control problem must be repeatedly solved.

Acknowledgements

Financial support received from the Spanish Ministry of Economy and Sustainability, and the European Regional Development Fund, through projects TOLERANT (DPI2006-05673), EHMAN (DPI2009-09386) and SIGERA (DPI2012-37154-C02-01), and the Generalitat de Catalunya (2009SGR-861) is thankfully acknowledged.

References

[1] Banga JB, Julio R, Balsa Canto E, Moles CG, Alonso AA. Dynamic optimization of bioprocesses: Efficient and robust numerical strategies. Journal of Biotechnology 2005;117(4):407-419.

[2] Biegler LT. An overview of simultaneous strategies for dynamic optimization. Chemical Engineering and Processing: Process Intensification 2007;46(11):1043-1053.

[3] Caballero JA, Grossmann IE. An algorithm for the use of surrogate models in modular flowsheet optimization. AICHE 2008;54:2633-2650.

[4] Carrasco EF, Banga JB. Dynamic Optimization of Batch Reactors Using Adaptive Stochastic Algorithms. Ind. Eng. Chem. Res 1997;36:2252-2261.

[5] Conti S, Gosling JP, Oakley J, O'Hagan A. Gaussian process emulation of dynamic computer codes. Biometrika 2009;96:663-676.

[6] DADEBO SA, MCAULEY KB. Dynamic optimization of constrained chemical engineering problems using dynamic programming. Computers & Chemical Engineering 1995;19:513-525.

[7] Diehl M, Bock HG, Diedam H, Wiebe P-B. Fast Direct Multiple Shooting Algorithms for Optimal Robot Control. Lecture Notes in Control and Information Sciences 2006;340: 65-93.

[8] Fang K-T, Li R, Sudjianto A. Design and modelling for computer experment. New York: Chapman and Hall/CRC; 2006.

[9] Forrester AIJ, Keane AJ. Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 2009;45:50-79.
[10]Forrester A, Sobester A, Keane A. Engineering design via surrogate modelling. Southampton, UK: John Wiley and Sons; 2008.
[11]Luus R. Optimal control of batch reactors by iterative dynamic programming. Journal of Process Control 1994;4(4):218-226.
[12]Meckesheimer M, Booker AJ, Barton RR, Simpson TW. Computationally inexpensive metamodel assessment strategies. American Institute of Aeronautics and Astronautics Journal 2002;40:2053-2060.
[13]Nagy ZK. Model based control of a yeast fermentation bioreactor using optimally designed artificial neural networks. Chemical Engineering Journal 2007;127:95-109.
[14]Srinivasana B, Palankib S, Bonvin D. Dynamic optimization of batch processes: I. Characterization of the nominal solution. Computers and Chemical Engineering 2003;27:1-26.